

REMARKS

This letter is responsive to the Final Office Action mailed on February 24, 2004 in which the Examiner maintained his rejection of claims 1-33 over Applicants' arguments submitted in a first response dated December 11, 2003. Reconsideration of the application is respectfully requested in light of the claim amendments and the following remarks.

Claim Amendments

The Applicants respectfully traverse the Examiner's rejections and the Examiner's counter arguments presented in the Final Office Action. However, Applicants have amended some of the claims to advance the prosecution.

A review of the Examiner's rejection indicates that the Examiner believes the claims are broadly written and could be construed as reading on methods of tracing pointers within the heap through the segregation of the heap into multiple trains, each with its own remembered set of pointers. The claims are amended herein, where necessary, to more clearly point out that the compact garbage collection tables are directed to the determination of a root set of live pointers into the heap from a call stack. As described in the specification, the call stack consists of a sequence of stack frames, where each stack frame in turn corresponds to a procedure whose execution has been suspended while a call in the body of the procedure is executed. The call in the body of the procedure is identified as a call site. The call stack must be traversed during a garbage collection scan to identify root set pointers in the call stack.

Claims 1, 11, 14, 18, 19, 24, 29, and 31 are amended herein to more distinctly claim that the descriptors in the descriptor table are used to identify locations of pointers from a call stack to a heap that are part of the root set of pointers used by the garbage collector when initiating a scan. In addition, detail has been added identifying that each descriptor describes stack locations within a stack frame (via an offset relative to a specified point in the stack frame) and registers that are associated with that stopping point and that contain heap pointers as described on page 8, lines 11-20 of the specification.

Claims 3, 13, 16, 26, 30 and 33 are canceled herein.

Rejections under 35 U.S.C. 103(a)

The Examiner maintains his rejection of the independent claims (i.e., 1, 11, 14, 24, 29 and 31) under 35 U.S.C. §103(a) as being unpatentable over Bush et al. (U.S. Pat. No. 6,308,319, hereinafter "Bush") in view of A. T. Garthwaite (U.S. Pat. No. 6,185,581, hereinafter "Garthwaite").

Bush is related to a "mechanism by which threads of a mutator can be efficiently suspended at safe points." Bush, Col. 3, lines 38-40. Bush discloses using a root set (See Bush, Col. 1, lines 63-66). In Bush, garbage collection is performed using a "'root set' of pointers to dynamically allocated memory locations" and starts by tracing the pointers of the root set into objects in the heap. Once in the heap, pointers in those objects are traced other objects in the heap. Bush is silent on the details of identifying the root set and there is no indication that Bush teaches or suggests any method of identifying the root set other than the monolithic root set described in the Background of Applicants' specification.

Bush certainly does not disclose descriptors that describe a set of registers within an unidentified stack frame of a call stack containing pointers to the heap and a set of offsets relative to a location within the unidentified stack frame as now claimed. Nor does Bush disclose identifying a root set through the compact garbage collection tables as now claimed.

Garthwaite is directed to mechanism for tracing pointers between objects within a heap by tracking the popularity of objects (how many pointers there are to objects in the heap from other objects in the heap) and combining popular objects together in generations, wherein the generations are further subdivided into "trains" having "cars". See Garthwaite abstract and Col. 13, lines 51-61. Garthwaite, in his background, discloses a root set as consisting "of reference values stored in the mutator's threads' call stacks, the CPU registers, and global variables outside the garbage-collected heap" (Garthwaite Col. 5, lines 42-44). See also Garthwaite FIGS. 4 and 5, and Col. 5, lines 49-63. Garthwaite does not teach how to identify root set pointers on call stacks that point to the heap. Rather, Garthwaite teaches how to track pointers located on the heap consisting of intergenerational pointers. In Garthwaite, after following pointers from the call into the heap, garbage collection requires that each generation have its own root set and that "pointers must be traced not only from the basic root set 52 [of each generation] but also from objects within the other generations." Garthwaite, Col 6, lines 63-65. Garthwaite teaches replacing the monolithic root set with a separate root set table for each generation in the heap and additional sets of intergenerational pointers.

While the applicants still believe that the rejection by the Examiner of the claims as filed was not sufficient to establish a prima facie case of obviousness, in order to advance prosecution, Applicants have amended the claims where necessary to more clearly point out that the set of compact garbage collection tables together identify root set pointers in a call stack for use by a garbage collector when initiating a garbage collection scan. Applicants feel that the amendments render the claims patentable for at least the following reasons in addition to those presented in the previous Response.

None of the references teach or suggest a set of three tables including a table of unique descriptors that are used to identify root set pointers on a call stack for use in garbage collection.

There is no teaching in Bush or Garthwaite of: a table storing a set of unique descriptors each unique descriptor describing a set of registers containing pointers to the heap and a set of offsets relative to a location within a stack frame containing pointers to the heap as now claimed in claims 1, 14 and 29; a set of registers to be associated with at least one stack frame of a call stack, the set of registers containing pointers to the heap, and a set of offsets relative to a location within the at least one stack frame containing pointers to the heap as claimed in claim 11; or accessing one of a set of unique descriptors in a descriptor table and describing a set of registers associated with a stack frame of a call stack and a set of offsets relative to a location within the stack frame containing pointers to the heap as claimed in claims 24 and 31.

The Examiner again cites Bush FIG. 1 and Col. 4, lines 18-25 as disclosing generating a final descriptor table storing a set of unique descriptors. The citation is provided below:

A portion of address space 101 is dynamically allocable as heap 102. Individual memory objects (e.g., objects 150, 151, 131, 132, 133, 134 and 141) are dynamically allocated from heap 102 and, in the referencing graph of FIG. 1, are currently reachable by a set of pointers including local variable L1, external or outer variable E1, and contents of registers R3 and R6 of register context 120. Bush, Col. 4, lines 18-25.

The above section describes the locations of pointers, not descriptors identifying the locations of pointers. There is no reference to a table of descriptors and certainly no reference to a table of unique descriptors describing a set of registers within a stack frame of a call stack containing pointers to the heap and a set of offsets relative to a location within the stack frame.

An inspection of FIG. 1 shows a standard computing environment of memory objects including a stack and registers with pointers into a heap. None of the elements described by Bush in this section or any other are required to store a set of unique descriptors as now defined. Thus, the Bush reference does not anticipate the final descriptor table as claimed in claims 1, 11, 14, 24, 29 and 31.

Nor can disclosure of a table having a set of unique descriptors be found in Garthwaite. While Garthwaite is rife with tables of references identifying the locations of pointers between generations, trains and cars of the heap, nothing in Garthwaite indicates that the tables consist of descriptors that are used to identify root set pointers on the call stack while traversing the call stack. Rather, Garthwaite discloses monolithic tables having references wherein each reference is associated with a car.

Garthwaite does not disclose a descriptor reference table as claimed

In the Final Office Action, the Examiner again cites Garthwaite, although to a previously uncited section, as disclosing generating "a descriptor reference table mapping a call site identifier in the first call site table to one of the unique descriptors in the final descriptor table." The citation is provided in pertinent part below:

"To identify references into the car from outside of it, train-algorithm implementations typically employ "remembered sets." As card tables are, remembered sets are used to keep track of references. Whereas a card-table entry contains information about references that the associated card contains, though, a remembered set associated with a given region contains information about references into that region from locations outside of it. In the case of the train algorithm, remembered sets are associated with car sections. Each remembered set, such as car 75's remembered set 90, lists locations in the generation that contain references into the associated car section. The remembered sets for all of a generation's cars are typically updated at the start of each collection cycle, concurrently with card-table updates. For reasons that will become apparent, the collector will typically not bother to place in the remembered set the locations of references from objects in car sections farther forward in the collection queue, i.e., from objects in older trains or cars added earlier to the same train. For the sake of simplicity, we will continue the assumption that only a single car is collected during each collection cycle, although we will discuss multiple-car collection sets presently.

When the remembered sets have been updated, the collector reads the remembered set of each car in the collection set to determine the location of each reference from a higher-order car into the collection set. The collector places the address of each reference thereby

found into a scratch-pad list associated with the train that contains that reference. (Again, the remembered set lists only reference locations in the same generation). Garthwaite, Col 8, line 64 – Col. 9, line 26.

Garthwaite's remembered sets contain reference locations and each remembered set is associated with a car. Garthwaite's remembered sets do not contain entries that map call site identifiers in a call site table to descriptors in a descriptor table because such indirect mapping would be redundant. Nothing in the cited section of Garthwaite suggests a table whose purpose is to map entries in a table of call sites in stack frames to a table of unique descriptors that describe registers to be associated with the stack frame and offsets within the stack frame.

Conclusion

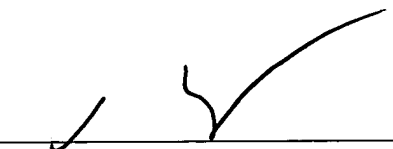
For the reasons cited above, Applicants believe that neither Bush nor Garthwaite alone or in combination anticipate all of the limitations of independent claims, as amended, in the pending application. Therefore, Applicants also believe that all of the claims in the present application are in a condition for allowance and that the Examiner's rejection of the dependent claims is therefore moot.

Claims 1, 2, 4-12, 14, 17-25, 27-29, and 31-32 remain pending in this application. This response is believed to be fully responsive to all points in the Office Action. The pending claims are believed to be in a condition for allowance. In view of the amendments and above remarks, Applicant respectfully requests a Notice of Allowance. If the Examiner believes a telephone conference would advance the prosecution of this Application, the Examiner is invited to telephone the undersigned at the below-listed telephone number.

Respectfully submitted,

Date: April 26, 2004

Customer No.: 23552



George C. Lewis, Reg. No. 53,214
Merchant & Gould P.C.
P.O. Box 2903
Minneapolis, MN 55402-0903
(303) 357-1639
(303) 357-1671 (fax)